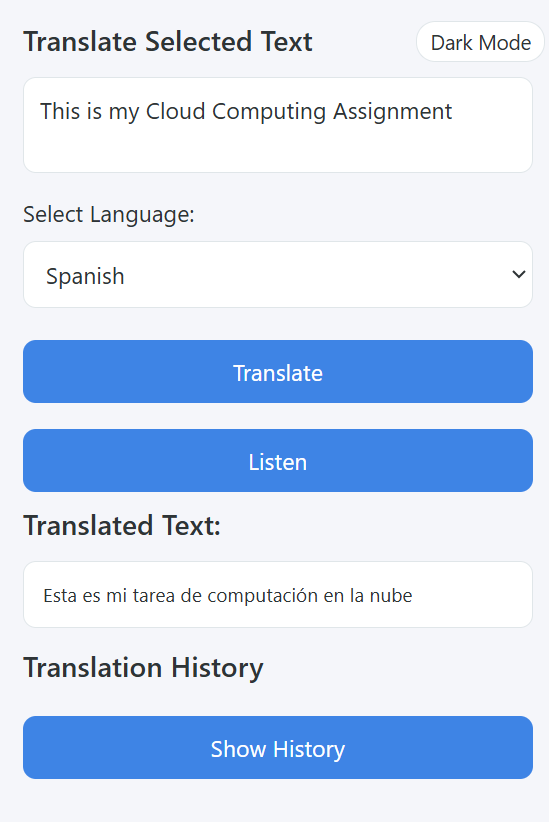
# 1. Introduction

## 1.1 Project Overview

The **AWS Translate & TTS Extension** is a browser extension that enables users to translate selected text and convert it into speech using AWS services. The extension provides a seamless experience for users by integrating **AWS Translate** for language translation and **AWS Polly** for text-to-speech conversion.

Additionally, an **Admin Panel** is implemented to monitor translation usage statistics and history. The admin panel fetches data from AWS APIs and presents it in an easy-to-use dashboard with dark mode support.

 A screenshot of a black screen

AI-generated content may be incorrect.

A screenshot of a computer

AI-generated content may be incorrect.

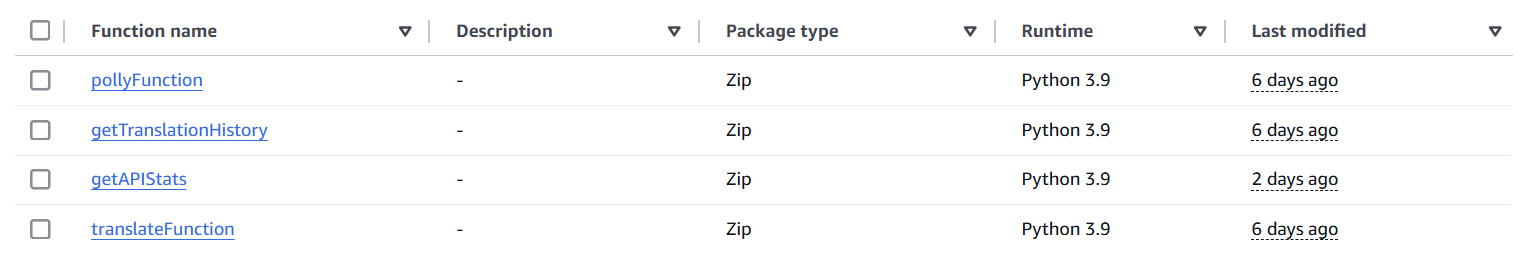
A screenshot of a computer

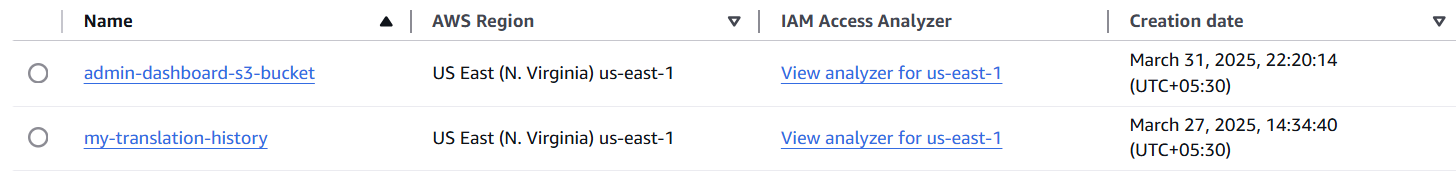
AI-generated content may be incorrect.

## 1.2 Cloud Services Used

* **AWS Translate** – Enables real-time text translation across multiple languages.
* **AWS Polly** – Converts translated text into natural-sounding speech.
* **AWS Lambda & API Gateway** – Used to create API endpoints for statistics and translation history retrieval.

Lamda Functions:



S3 Bucke

# 2. System Architecture

## 2.1 Workflow Overview

1. The user selects text on a webpage.
2. The extension captures the text and sends it to AWS Translate.
3. The translated text is retrieved and displayed in the extension popup.
4. If the user chooses, AWS Polly converts the translated text into speech.
5. The admin panel fetches translation history and usage statistics from AWS for monitoring.

## 2.2 Architecture Diagram

A diagram of a cloud service

AI-generated content may be incorrect.

# 3. Implementation Details

## 3.1 Browser Extension

### 3.1.1 Manifest Configuration (manifest.json)

* Specifies extension permissions, such as activeTab, storage, and API access.
* Defines background scripts and content scripts.

### 3.1.2 Content Script (content.js)

* Listens for user-selected text.
* Sends the selected text to the background script.

### 3.1.3 Background Script (background.js)

* Handles API calls to AWS Translate and AWS Polly.
* Sends back the translated text and audio URL to the popup script.
* Manages error handling and retries.

### 3.1.4 Popup UI (popup.html, popup.js, styles.css)

* Provides a user interface for displaying translations and playing TTS audio.
* Includes dark mode support.
* Displays translation history.

## 3.2 Admin Panel

### 3.2.1 Dashboard (index.html, dashboard.js, style.css)

* **User Interface:**
  + Displays total translations and top-used languages.
  + Provides a table of recent translation history.
* **API Integration:**
  + Fetches statistics (STATS\_API) and history (HISTORY\_API) from AWS.
  + Implements retry logic for API failures.
* **Theme Toggle:**
  + Allows users to switch between light and dark modes.

### 3.2.2 Backend APIs

* **STATS\_API (Fetches translation statistics)**
* **HISTORY\_API (Fetches translation history with timestamps)**
* **Retry Mechanism:** Ensures reliable data retrieval by retrying API calls up to three times in case of failure.

# 4. AWS Integration

## 4.1 API Calls to AWS Services

* **AWS Translate API:**
  + Converts selected text from the source language to the target language.
  + Example request and response structure.
* **AWS Polly API:**
  + Converts translated text into speech.
  + Returns an audio URL for playback.

## 4.2 Security & Permissions

* Ensures minimal permissions for the extension.
* Uses secure API keys and authentication mechanisms.
* Limits access to sensitive data.

# 5. Conclusion

## 5.1 Summary of Implementation

* A browser extension was developed using AWS Translate and AWS Polly to provide text translation and text-to-speech conversion.
* An admin panel was created to monitor translation usage and history.
* AWS Lambda and API Gateway were used to fetch data for the admin panel.
* Security measures were implemented to ensure data integrity.

## 5.2 Key Takeaways

* Integration with cloud services like AWS enhances the functionality of browser extensions.
* Using an admin panel improves monitoring and data visualization.
* Retry mechanisms improve API reliability.